REST USING SPRING FRAMEWORK

# Introduction

You can use Spring framework to define your REST implementation. The following annotations are used.

**@Controller** – Used in the class level

**@RequestMapping("/emp")** – Used in the class level as the root path

**@RequestMapping(value = "ID/{id}", method = RequestMethod.*GET*)** - Used in method level as the sub path

**@ResponseBody** – Used in the method leve

**@PathVariable** String id – Used in the method parameter

**@RequestParam("id")** String id – used in the method parameter to pass id as request parameter.

**@RequestMapping(value = "create", method = RequestMethod.*POST*,**

**consumes = {MediaType.*APPLICATION\_JSON\_VALUE*, MediaType.*APPLICATION\_XML\_VALUE* },**

**produces = {MediaType.*APPLICATION\_JSON\_VALUE*, MediaType.*APPLICATION\_XML\_VALUE* })**

The above annotation defines that the server can accept both JSON and XML request as request body, to specify you to provide the value as follows.

**consumes = {MediaType.*APPLICATION\_JSON\_VALUE*, MediaType.*APPLICATION\_XML\_VALUE* }**

Similarly it provides information to produce the result in the XML or JSON format, to specify you have to give the values as

**produces = {MediaType.*APPLICATION\_JSON\_VALUE*, MediaType.*APPLICATION\_XML\_VALUE* }**

**@ResponseStatus(HttpStatus.*CREATED*)** – Used in the method level as Created

**@ResponseBody** - spring will try to convert its return value and write it to the http response automatically

**@RequestBody Employee emp** - Used in method parameter level. Spring will try to convert the content of the incoming request body to your parameter object on the fly.

Remember that Spring controller can accept both XML or JSON request body or either or.

The complete project is given below. The name of the project “restspring” . The following jar files are used.

**aopalliance-1.0.jar**

**commons-logging-1.1.1.jar**

**jackson-core-asl-1.9.10.jar**

**jackson-mapper-asl-1.9.10.jar**

**servlet-api-2.5.jar**

**spring-aop-3.2.2.RELEASE.jar**

**spring-beans-3.2.2.RELEASE.jar**

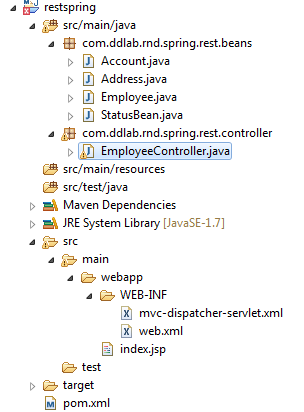
**spring-context-3.2.2.RELEASE.jar**

**spring-core-3.2.2.RELEASE.jar**

**spring-expression-3.2.2.RELEASE.jar**

**spring-web-3.2.2.RELEASE.jar**

**spring-webmvc-3.2.2.RELEASE.jar**



The following beans are given below.

**Account.java**

package com.ddlab.rnd.spring.rest.beans;

import javax.xml.bind.annotation.XmlElement;

import javax.xml.bind.annotation.XmlRootElement;

import org.codehaus.jackson.annotate.JsonProperty;

@XmlRootElement

public class Account {

@JsonProperty(value="AccountName") private String name;

@JsonProperty(value="AccountNo") private String actNo;

@XmlElement

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@XmlElement

public String getActNo() {

return actNo;

}

public void setActNo(String actNo) {

this.actNo = actNo;

}

@Override

public int hashCode() {

final int prime = 31;

int result = 1;

result = prime \* result + ((actNo == null) ? 0 : actNo.hashCode());

result = prime \* result + ((name == null) ? 0 : name.hashCode());

return result;

}

@Override

public boolean equals(Object obj) {

if (this == obj)

return true;

if (obj == null)

return false;

if (getClass() != obj.getClass())

return false;

Account other = (Account) obj;

if (actNo == null) {

if (other.actNo != null)

return false;

} else if (!actNo.equals(other.actNo))

return false;

if (name == null) {

if (other.name != null)

return false;

} else if (!name.equals(other.name))

return false;

return true;

}

@Override

public String toString() {

return "Account [name=" + name + ", actNo=" + actNo + "]";

}

}

**Address.java**

package com.ddlab.rnd.spring.rest.beans;

import javax.xml.bind.annotation.XmlElement;

import javax.xml.bind.annotation.XmlRootElement;

import org.codehaus.jackson.annotate.JsonProperty;

@XmlRootElement

public class Address {

@JsonProperty(value="StreetName") private String streetName;

@JsonProperty(value="CountryName") private String countryName;

@XmlElement

public String getStreetName() {

return streetName;

}

public void setStreetName(String streetName) {

this.streetName = streetName;

}

@XmlElement

public String getCountryName() {

return countryName;

}

public void setCountryName(String countryName) {

this.countryName = countryName;

}

@Override

public int hashCode() {

final int prime = 31;

int result = 1;

result = prime \* result

+ ((countryName == null) ? 0 : countryName.hashCode());

result = prime \* result

+ ((streetName == null) ? 0 : streetName.hashCode());

return result;

}

@Override

public boolean equals(Object obj) {

if (this == obj)

return true;

if (obj == null)

return false;

if (getClass() != obj.getClass())

return false;

Address other = (Address) obj;

if (countryName == null) {

if (other.countryName != null)

return false;

} else if (!countryName.equals(other.countryName))

return false;

if (streetName == null) {

if (other.streetName != null)

return false;

} else if (!streetName.equals(other.streetName))

return false;

return true;

}

@Override

public String toString() {

return "Address [streetName=" + streetName + ", countryName="

+ countryName + "]";

}

}

**Employee.java**

package com.ddlab.rnd.spring.rest.beans;

import javax.xml.bind.annotation.XmlElement;

import javax.xml.bind.annotation.XmlRootElement;

import org.codehaus.jackson.annotate.JsonProperty;

import org.codehaus.jackson.map.annotate.JsonRootName;

@XmlRootElement

@JsonRootName(value="Emp")

public class Employee {

@JsonProperty(value="Id") private String id;

@JsonProperty(value="Name") private String name;

@JsonProperty(value="Adrs") private Address adrs;

@JsonProperty(value="Account") private Account act;

public Employee( String id )

{

this.id = id;

}

public Employee()

{

}

@XmlElement

public Address getAdrs() {

return adrs;

}

public void setAdrs(Address adrs) {

this.adrs = adrs;

}

@XmlElement

public Account getAct() {

return act;

}

public void setAct(Account act) {

this.act = act;

}

@XmlElement

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

@XmlElement

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public int hashCode() {

final int prime = 31;

int result = 1;

result = prime \* result + ((id == null) ? 0 : id.hashCode());

return result;

}

@Override

public boolean equals(Object obj) {

if (this == obj)

return true;

if (obj == null)

return false;

if (getClass() != obj.getClass())

return false;

Employee other = (Employee) obj;

if (id == null) {

if (other.id != null)

return false;

} else if (!id.equals(other.id))

return false;

return true;

}

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", adrs=" + adrs

+ ", act=" + act + "]";

}

}

**Status.java**

package com.ddlab.rnd.spring.rest.beans;

import javax.xml.bind.annotation.XmlElement;

import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement(name="Status")

public class StatusBean {

private String status = "Failed";

private String errCode = "";

private String errMsg = "";

@XmlElement(defaultValue="Status")

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

@XmlElement(defaultValue="ErrorCode")

public String getErrCode() {

return errCode;

}

public void setErrCode(String errCode) {

this.errCode = errCode;

}

@XmlElement(defaultValue="ErrorMessage")

public String getErrMsg() {

return errMsg;

}

public void setErrMsg(String errMsg) {

this.errMsg = errMsg;

}

}

**EmployeeController.java**

package com.ddlab.rnd.spring.rest.controller;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Enumeration;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.http.HttpStatus;

import org.springframework.http.MediaType;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.bind.annotation.ResponseStatus;

import org.springframework.web.context.request.RequestContextHolder;

import org.springframework.web.context.request.ServletRequestAttributes;

import com.ddlab.rnd.spring.rest.beans.Employee;

import com.ddlab.rnd.spring.rest.beans.StatusBean;

@Controller

@RequestMapping("/emp")

public class EmployeeController {

private List<Employee> empList = new ArrayList<Employee>();

@RequestMapping(value = "ID/{id}", method = RequestMethod.GET)

@ResponseBody

public Object getEmp( @PathVariable String id )

{

Object obj = null;

Employee emp = new Employee(id);

if( empList.contains(emp))

{

for( Employee emp1 : empList )

{

if( emp.equals(emp1))

{

obj = emp1;

break;

}

}

}

else

{

StatusBean statusBean = new StatusBean();

statusBean.setErrCode("002");

statusBean.setErrMsg("Employee with this id "+id+" does not exist");

obj = statusBean;

}

return obj;

}

@RequestMapping(value = "ID", method = RequestMethod.GET)

@ResponseBody

public Object getEmp1( @RequestParam("id") String id )

{

/\*Request will be like this

\* http://localhost:8080/restspring/emp/ID?id=Id123

\*/

System.out.println("------- Invoking method getEmp1 with @RequestParam as parameter ------");

Object obj = null;

Employee emp = new Employee(id);

if( empList.contains(emp))

{

for( Employee emp1 : empList )

{

if( emp.equals(emp1))

{

obj = emp1;

break;

}

}

}

else

{

StatusBean statusBean = new StatusBean();

statusBean.setErrCode("002");

statusBean.setErrMsg("Employee with this id "+id+" does not exist");

obj = statusBean;

}

return obj;

}

@RequestMapping(value = "create", method = RequestMethod.POST,

consumes ={MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE }, produces = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

@ResponseStatus(HttpStatus.CREATED)

@ResponseBody

public StatusBean create( @RequestBody Employee emp ) {

StatusBean statusBean = new StatusBean();

if( empList.contains(emp) )

{

statusBean.setErrCode("001");

statusBean.setErrMsg("Employee with this id "+emp.getId()+" already exists");

}

else

{

empList.add(emp);

statusBean.setStatus("Success");

}

display();

return statusBean;

}

@RequestMapping(value = "create1", method = RequestMethod.POST,

consumes = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE }, produces = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

@ResponseStatus(HttpStatus.CREATED)

@ResponseBody

public StatusBean create1( @RequestBody Employee emp ) {

/\*

\* Get all the request header information

\* Request will be http://localhost:8080/restspring/emp/create1

\*/

System.out.println("-------- Inside Method create1 with no HttpServletRequest Object -------");

ServletRequestAttributes attr = (ServletRequestAttributes) RequestContextHolder.currentRequestAttributes();

HttpServletRequest request = attr.getRequest();

Enumeration en = request.getHeaderNames();

while( en.hasMoreElements() ) {

Object obj = en.nextElement();

System.out.println("Header Name : "+obj);

System.out.println("Header Value : "+request.getHeader((String)obj));

}

StatusBean statusBean = new StatusBean();

if( empList.contains(emp) )

{

statusBean.setErrCode("001");

statusBean.setErrMsg("Employee with this id "+emp.getId()+" already exists");

}

else

{

empList.add(emp);

statusBean.setStatus("Success");

}

return statusBean;

}

@RequestMapping(value = "create2", method = RequestMethod.POST,

consumes = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE }, produces = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

@ResponseStatus(HttpStatus.CREATED)

@ResponseBody

public StatusBean create2( @RequestBody Employee emp , HttpServletRequest request) {

/\*

\* Get all the request header information

\* Request will be http://localhost:8080/restspring/emp/create2

\*/

System.out.println("-------- Inside Method create2 with HttpServletRequest Object as method parameter -------");

Enumeration en = request.getHeaderNames();

while( en.hasMoreElements() ) {

Object obj = en.nextElement();

System.out.println("Header Name : "+obj);

System.out.println("Header Value : "+request.getHeader((String)obj));

}

StatusBean statusBean = new StatusBean();

if( empList.contains(emp) )

{

statusBean.setErrCode("001");

statusBean.setErrMsg("Employee with this id "+emp.getId()+" already exists");

}

else

{

empList.add(emp);

statusBean.setStatus("Success");

}

return statusBean;

}

@RequestMapping(value = "create3", method = RequestMethod.POST,

consumes = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE }, produces = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

@ResponseStatus(HttpStatus.CREATED)

@ResponseBody

public StatusBean create3( @RequestBody Employee emp , HttpServletRequest request , HttpServletResponse response ) {

/\*

\* Get all the request header information

\* Request will be http://localhost:8080/restspring/emp/create3

\*/

System.out.println("-------- Inside Method create3 with HttpServletResponse Object as method parameter -------");

System.out.println("Response Object : "+response);

Enumeration en = request.getHeaderNames();

while( en.hasMoreElements() ) {

Object obj = en.nextElement();

System.out.println("Header Name : "+obj);

System.out.println("Header Value : "+request.getHeader((String)obj));

}

StatusBean statusBean = new StatusBean();

if( empList.contains(emp) )

{

statusBean.setErrCode("001");

statusBean.setErrMsg("Employee with this id "+emp.getId()+" already exists");

}

else

{

empList.add(emp);

statusBean.setStatus("Success");

}

try {

//Send a response error message

response.sendError(HttpServletResponse.SC\_SERVICE\_UNAVAILABLE,"This service is not available for you");

} catch (IOException e) {

e.printStackTrace();

}

return statusBean;

}

@RequestMapping(value = "update", method = RequestMethod.PUT,

consumes = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE }, produces = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

@ResponseStatus( HttpStatus.CREATED )

@ResponseBody

public StatusBean update( @RequestBody Employee emp ) {

StatusBean statusBean = new StatusBean();

if( empList.contains(emp))

{

for( Employee emp1 : empList )

{

if( emp.equals(emp1))

{

empList.remove(emp1);

empList.add(emp);

statusBean.setStatus("Success");

break;

}

}

}

else

{

statusBean.setErrCode("002");

statusBean.setErrMsg("Employee with this id "+emp.getId()+" does not exist");

}

return statusBean;

}

@RequestMapping(value = "delete", method = RequestMethod.DELETE,

consumes = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE }, produces = {MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

@ResponseStatus( HttpStatus.CREATED )

@ResponseBody

public StatusBean delete( @RequestBody Employee emp ) {

StatusBean statusBean = new StatusBean();

if( empList.contains(emp))

{

for( Employee emp1 : empList )

{

if( emp.equals(emp1))

{

empList.remove(emp1);

statusBean.setStatus("Success");

break;

}

}

}

else

{

statusBean.setErrCode("003");

statusBean.setErrMsg("Employee with this id "+emp.getId()+" does not exist, hence data cannot be deleted");

}

return statusBean;

}

public void display()

{

for( Employee emps : empList )

System.out.println(emps);

System.out.println("-----------------\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*---------------------");

}

}

**WEB-INF/mvc-dispatcher-servlet.xml**

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:mvc=*"http://www.springframework.org/schema/mvc"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"*

*http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-3.0.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context-3.0.xsd*

*http://www.springframework.org/schema/mvc*

*http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd"*>

<context:component-scan base-package=*"com.ddlab.rnd.spring.rest"* />

<mvc:annotation-driven />

</beans>

**web.xml**

<web-app id=*"WebApp\_ID"* version=*"2.4"*

xmlns=*"http://java.sun.com/xml/ns/j2ee"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://java.sun.com/xml/ns/j2ee*

*http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd"*>

<display-name>Spring Web MVC Application</display-name>

<servlet>

<servlet-name>mvc-dispatcher</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>mvc-dispatcher</servlet-name>

<url-pattern>/\*</url-pattern>

</servlet-mapping>

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>/WEB-INF/mvc-dispatcher-servlet.xml</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

</web-app>

**Pom.xml**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>restspring</groupId>

<artifactId>restspring</artifactId>

<packaging>war</packaging>

<version>0.0.1-SNAPSHOT</version>

<name>restspring Maven Webapp</name>

<url>http://maven.apache.org</url>

<properties>

<spring.version>3.2.2.RELEASE</spring.version>

<jackson.version>1.9.10</jackson.version>

<jdk.version>1.6</jdk.version>

</properties>

<dependencies>

<!-- Spring 3 dependencies -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- Jackson JSON Mapper -->

<dependency>

<groupId>org.codehaus.jackson</groupId>

<artifactId>jackson-mapper-asl</artifactId>

<version>${jackson.version}</version>

</dependency>

<!-- Servlet -API -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

<version>2.5</version>

<!-- <scope>provided</scope> -->

</dependency>

</dependencies>

<build>

<finalName>restspring</finalName>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>2.3.2</version>

<configuration>

<source>${jdk.version}</source>

<target>${jdk.version}</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

# Testing

**GET Request as XML**

http://localhost:8080/restspring/emp/ID/Id1

In the header, provide

**Accept : Application/xml**

**Output**

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Status>

<errCode>002</errCode>

<errMsg>Employee with this id Id1 does not exist</errMsg>

<status>Failed</status>

</Status>

**GET Request as JSON**

http://localhost:8080/restspring/emp/ID/Id1

In the header, provide

**Accept : Application/json**

**Output**

{

"status": "Failed",

"errCode": "002",

"errMsg": "Employee with this id Id1 does not exist"

}

**POST Request as XML**

http://localhost:8080/restspring/emp/create

**Content-type : application/xml**

**Accept : application/xml**

**XML Request**

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<employee>

<act>

<actNo>11111</actNo>

<name>DD Mishra</name>

</act>

<adrs>

<countryName>India</countryName>

<streetName>Bangalore</streetName>

</adrs>

<id>Id1</id>

<name>Name-1</name>

</employee>

**Output**

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Status>

<errCode></errCode>

<errMsg></errMsg>

<status>Success</status>

</Status>

**POST Request as JSON**

http://localhost:8080/restspring/emp/create

**Content-type : application/json**

**Accept : application/json**

**JSON Request**

{

"Id" : "Id5",

"Name" : "Name-1",

"Adrs" : {

"StreetName" : "Bangalore",

"CountryName" : "India"

},

"Account" : {

"AccountName" : "DD Mishra",

"AccountNo" : "11111"

}

}

**Output**

{

"status": "Success",

"errCode": "",

"errMsg": ""

}

**PUT Request as XML**

http://localhost:8080/springrest1/emp/update

**Content-type : application/json**

**Accept : application/json**

**XML Request**

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<employee>

<act>

<actNo>11111</actNo>

<name>Deba Mishra</name>

</act>

<adrs>

<countryName>India</countryName>

<streetName>Bhubaneswar</streetName>

</adrs>

<id>Id1</id>

<name>Name-1</name>

</employee>

**Output**

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Status>

<errCode></errCode>

<errMsg></errMsg>

<status>Success</status>

</Status>

PUT Request as JSON

--------------------

http://localhost:8080/springrest1/emp/update

**Content-type : application/json**

**Accept : application/json**

**JSOn Request**

{

"Id" : "Id1",

"Name" : "Debadatta Mishra",

"Adrs" : {

"StreetName" : "Odisha",

"CountryName" : "India"

},

"Account" : {

"AccountName" : "Debadatta Mishra",

"AccountNo" : "11111"

}

}

**Output**

{

"status": "Success",

"errCode": "",

"errMsg": ""

}

**Delete Request as XML**

http://localhost:8080/springrest1/emp/delete

**Content-Type : application/xml**

**Accept : application/xml**

**XML Request**

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<employee>

<act>

<actNo>11111</actNo>

<name>Deba Mishra</name>

</act>

<adrs>

<countryName>India</countryName>

<streetName>Bangalore</streetName>

</adrs>

<id>Id1</id>

<name>Name-1</name>

</employee>

**Output**

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Status>

<errCode></errCode>

<errMsg></errMsg>

<status>Success</status>

</Status>

**Delete Request as JSON**

http://localhost:8080/springrest1/emp/delete

Content-Type : application/json

Accept : application/json

**JSON Request**

{

"Id" : "Id2",

"Name" : "Debadatta Mishra",

"Adrs" : {

"StreetName" : "Odisha",

"CountryName" : "India"

},

"Account" : {

"AccountName" : "Debadatta Mishra",

"AccountNo" : "11111"

}

}

**Output**

{

"status": "Success",

"errCode": "",

"errMsg": ""

}

**Difference between ACCEPT and Content-Type**

ACCEPT is what the browser is able to digest

Content-Type is what format the actual data is in,

Content-Type is the format of the data to be sent to the server.

Accept is the format of the data sent by the server.